MOUSETRAP: Designing High-Speed Asynchronous Digital Pipelines

Montek Singh
Univ. of North Carolina, Chapel Hill, NC
(formerly at Columbia University)

Steven M. Nowick
Columbia University, New York, NY
Contribution

Pipeline style that is:

- **asynchronous**: avoids problems of high-speed global clock
- **very high-speed**
- **naturally elastic**: hold dynamically-variable # of data items
- **uses simple local timing constraints**: one-sided
- **robustly support variable-speed environments**
- **well-matched for fine-grain datapaths**

**Publications:**


M. Singh and S.M. Nowick, "Ultra-High-Speed Transition-Signaling Asynchronous Pipelines." Proc. of IEEE Int. Conf. on Computer Design (ICCD), Austin, TX (Sept. 2001)
MOUSETRAP Pipelines

Simple asynchronous implementation style, uses...

- *level-sensitive D-latches (not flipflops)*
- *simple stage controller:* 1 gate/pipeline stage
- *single-rail bundled data:* synchronous style logic blocks
  (1 wire/bit, with matched delay)

Target = static logic blocks

**Goal:** very fast cycle time

- simple inter-stage communication
“MOUSETRAP”: uses a “capture-pass protocol”

Latches ...:
- normally transparent: *before* new data arrives
- become opaque: *after* data arrives (= “capture” data)

Control Signaling: “*transition-signaling*” = 2-phase
- simple “req/ack” protocol = only 2 events per handshake (not 4)
- *no* “return-to-zero”
- each transition (up/down) signals a distinct operation
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

Data in → Stage N-1 → Data Latch → Stage N → Data Latch → Stage N+1 → Data out

1\textsuperscript{st} data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

1st data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

[Diagram showing stages communicating with Latch Controller]

1st data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using transition-signaling:

1 transition
per data item!

1st data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using transition-signaling:

1 transition per data item!

Data in → Stages communicate using transition-signaling:
→ Latch Controller
→ Stages communicate using transition-signaling:
→ Data Latch
→ Data out

1st data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling:*

1 transition per data item!

1st data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

1^{st} data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using \textit{transition-signaling}:

1 transition per data item!

Stage $N-1$  
Data Latch  
Stage $N$  
Latch Controller  
Stage $N+1$

Data in  
\[ req_N \quad done_N \quad ack_N \quad ack_{N-1} \quad req_{N+1} \]

Data out  
\[ En \]

1\textsuperscript{st} data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

2\textsuperscript{nd} data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

1st data item flowing through the pipeline

2nd data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using **transition-signaling:**

- **1 transition per data item!**
- **2nd data item flowing through the pipeline**
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

1st data item flowing through the pipeline

2nd data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using transition-signaling:

1 transition per data item!

Data in → Stage N-1 → Data Latch → Stage N → Latch Controller → Stage N+1 → Data out

2nd data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using transition-signaling:

1 transition per data item!

$1^{st}$ data item flowing through the pipeline

$2^{nd}$ data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

1st data item flowing through the pipeline

2nd data item flowing through the pipeline
MOUSETRAP: A Basic FIFO

Stages communicate using *transition-signaling*:

1 transition per data item!

2\textsuperscript{nd} data item flowing through the pipeline
Latch controller (XNOR) acts as “phase converter”:

- 2 distinct transitions (up or down) ➔ pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as "phase converter":

- 2 distinct transitions (up or down) \(\rightarrow\) pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as "phase converter":
- 2 distinct transitions (up or down) \(\Rightarrow\) *pulsed latch enable*

**2 transitions per latch cycle**

![Diagram](image.png)
MOUSETRAP: A Basic FIFO (contd.)

Latch controller (XNOR) acts as "phase converter":
- 2 distinct transitions (up or down) \( \Rightarrow \) pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as "phase converter":
- 2 distinct transitions (up or down) ➞ pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as “phase converter”:

- 2 distinct transitions (up or down)  \( \rightarrow \) pulsed latch enable

2 transitions per latch cycle

Mousetrap: A Basic FIFO (contd.)
Latch controller (XNOR) acts as “phase converter”:
- 2 distinct transitions (up or down) → pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as "phase converter":
- 2 distinct transitions (up or down) ➞ pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as "phase converter":
• 2 distinct transitions (up or down) \(\rightarrow\) pulsed latch enable

Latch is disabled when current stage is "done"
MOUSETRAP: A Basic FIFO (contd.)

Latch controller (XNOR) acts as "phase converter":
- 2 distinct transitions (up or down) → pulsed latch enable

2 transitions per latch cycle

Data Latch

Stage N-1

Stage N

Stage N+1

Data in

Data out

Req

Ack

En

Done

Req

Ack
Latch controller (XNOR) acts as “phase converter”:
- 2 distinct transitions (up or down) \(\Rightarrow\) pulsed latch enable

2 transitions per latch cycle
Latch controller (XNOR) acts as “phase converter”:
- 2 distinct transitions (up or down)  ➔ pulsed latch enable

2 transitions per latch cycle

MOUSETRAP: A Basic FIFO (contd.)
Latch controller (XNOR) acts as "phase converter":
- 2 distinct transitions (up or down) \( \Rightarrow \) pulsed latch enable

**2 transitions per latch cycle**

MOUSETRAP: A Basic FIFO (contd.)
MOUSETRAP: A Basic FIFO (contd.)

Latch controller (XNOR) acts as “phase converter”:

- 2 distinct transitions (up or down) → pulsed latch enable

```
<table>
<thead>
<tr>
<th>Data Latch</th>
<th>Latch Controller</th>
<th>Data in</th>
<th>Data out</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage N-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage N+1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

2 transitions per latch cycle
Latch controller (XNOR) acts as "phase converter":

- 2 distinct transitions (up or down) ➞ pulsed latch enable

Latch is re-enabled when next stage is "done"
Detailed Controller Operation

* One pulse per data item flowing through:
  - **down transition**: caused by “done” of N
  - **up transition**: caused by “done” of N+1

* No minimum pulse width constraint!
  - simply, down transition should start “early enough”
  - can be “negative width” (no pulse!)
MOUSETRAP: FIFO Cycle Time

![Diagram of FIFO cycle time with stages and control signals]
MOUSETRAP: FIFO Cycle Time

Data in → Stage N-1 → Stage N → Stage N+1 → Data out

Latch Controller

$\text{ack}_{N-1}$ → $\text{req}_N$ → $\text{En}$ → $\text{done}_N$ → $\text{ack}_N$ → $\text{req}_{N+1}$

$N$ computes
MOUSETRAP: FIFO Cycle Time

Fast self-loop: N disables itself
MOUSETRAP: FIFO Cycle Time

Data in → Stage N-1 → Data Latch (done_N) → Stage N → Data Latch (done_N) → Stage N+1 → Data out

- req_N
- ack_N
- req_N-1
- ack_N-1

N+1 computes
MOUSETRAP: FIFO Cycle Time

Stage N-1

Data Latch

Latch Controller

ack_{N-1}

req_{N}

done_{N}

Data in

Stage N

Data Latch

N re-enabled to compute

Stage N+1

Data out

ack_{N}

req_{N+1}
MOUSETRAP: FIFO Cycle Time

Cycle Time = 2 T_{LATCH} + T_{XNOR}
MOUSETRAP: Pipeline With Logic

Simple Extension to FIFO:
insert *logic block + matching delay* in each stage

Logic Blocks: can use *standard single-rail* (non-hazard-free)

“Bundling” Requirement:
- each “req” must arrive *after* data inputs valid and stable
Timing Analysis

Main Timing Constraint: avoid “data overrun”

Data must be safely “captured” by Stage N before new inputs arrive from Stage N-1

- Simple 1-sided timing constraint: fast latch disable
- Stage N’s “self-loop” faster than entire path through previous stage
Timing Analysis

Main Timing Constraint: avoid “data overrun”

Data must be safely “captured” by Stage N before new inputs arrive from Stage N-1

- simple 1-sided timing constraint: fast latch disable
- Stage N’s “self-loop” faster than entire path through previous stage

![Diagram](image_url)
Timing Analysis

Main Timing Constraint: avoid “data overrun”

Data must be safely “captured” by Stage N before new inputs arrive from Stage N-1

- simple 1-sided timing constraint: fast latch disable
- Stage N’s “self-loop” faster than entire path through previous stage
Timing Analysis

Main Timing Constraint: avoid “data overrun”

Data must be safely “captured” by Stage N before new inputs arrive from Stage N-1

- simple 1-sided timing constraint: fast latch disable
- Stage N’s “self-loop” faster than entire path through previous stage